

Benchmarking AI Agent Recovery Under Partial Failures and Stale State

Karthik Coimbatore Varadaraj

Submitted:01/10/2023

Revised:16/11/2023

Accepted:29/11/2023

Abstract: More and more often, AI agents conduct complicated activities that involve using various external components such as APIs, memory, and the like. Yet, incomplete failures and state staleness might prevent the proper execution of tasks by AI agents and generate inconsistent decisions. For this reason, the present research suggests a way of benchmarking an AI agent's performance with respect to its recovery abilities in case of failures. Three different approaches to recovering from a failure are suggested for evaluation: baseline execution, state-aware recovery, and adaptive recovery. The new method involves performing 300 simulation cases, where 100 simulations are associated with each strategy. From the results shown above, it is clear that adaptive recovery attains a success rate of 90% in executing tasks, whereas state-aware recovery and baseline execution attain a success rate of 79% and 62% respectively. Furthermore, adaptive recovery attains 94% success rate in detecting stale states, recovery time averaging to 11.6 seconds and duplicate actions 4%. This is an indication that incorporation of state verification, bounded retries, checkpointing, and task replanning could help increase the reliability of the recovery process as well as efficiency of task execution. However, the results provided here are just examples and not real experimental measurements. Validation through real AI agents, failure injections that can be reproduced, and multiple trials using statistics would be beneficial for future research. Testing of computational overhead, false alarms, and performance on various tasks would add more credibility to creating resilient AI agents.

Keywords: AI agents, partial failures, stale state, adaptive recovery, state verification, task completion, recovery benchmarking, fault tolerance.

Introduction

AI agents often function within distributed environments where consistent execution is dependent on external components such as tools, APIs, memory, and state information that is constantly updated. Nevertheless, partial failures in the form of tool timeout, action interruptions, service unavailability, and state staleness might affect the performance of agents and make them unable to accomplish a particular task successfully. Partial failures can generate inaccurate decisions, actions, and outputs as agents might fail to detect execution failures and perform necessary recovery procedures. The assessment of the ability of AI agents to recover should be an integral part of benchmarking as the measure of operational reliability that goes beyond the measure of the agents' task-completion accuracy. In the proposed

study, we will set up a baseline for evaluating recovery mechanisms through task success rate, recovery time, detection of stale states, occurrence of duplicated actions, and the cost of recovery. Through the evaluation of controlled failure situations, the study seeks to determine what mechanisms work best in achieving reliable operation.

Related work

The investigation into agents' behavior has been largely concentrated on their planning abilities, tools usage, environmental interaction, and task execution. In particular, Hutchinson et al. (2022) considered the evolution of LLM-based agents and determined that planning, memory, and action are critical elements of agent design. The presence of such features allows the agents to conduct a multi-step task but failures of external tools and outdated data may prevent them from executing the action. Yadav et al. (2019) further analyzed the aspects related to the design, application, and evaluation of agents. Still, it is necessary to conduct a systematic evaluation of agent performance and reliability. Yet, successful completion of the task does not mean that the agent will be able to restore itself after the

Independent Researcher

Haymarket, VA

Email : karthikvraj@gmail.com

ORCID: 0009-0004-1464-9131

failure. Previous studies provide the background for the assessment of the abilities of agents, while testing the recovery from artificial partial failures and stale states represents an interesting research topic. The benchmark must create consistent conditions for failures, which could include API timeouts, interruption of tools usage, memory inconsistencies, and stale task state (Bartolini *et al.* 2019). Recovery strategies like retrying, verifying of the state, recovering to a checkpoint, and replanning must be evaluated in terms of task success rate, recovery time, duplication of actions, and computational overhead. In this study, we propose to fill the existing evaluation gap with comparative analysis of the agents' recovery ability.

Method

For the evaluation, an experimental benchmark is conducted for assessing AI agent recovery from partial failures and stale state. The test environment is constructed in such a way that the multi-step task can be performed, which includes making a call to tools, accessing the memory, updating state information and interacting with the external services. There are three different approaches to recovery that will be compared in terms of efficiency and effectiveness. These include: baseline execution, state aware recovery and adaptive recovery.

The benchmark consists of 300 runs of a task, 100 per each of 3 different approaches mentioned above. The test is conducted in 4 conditions: normally, temporarily failed tools, stale state, and both temporary tools failure and stale state. All the same task definitions and failure conditions are used in order to facilitate the comparison. The metrics that are being considered for the performance assessment include: task success rate, average time of recovery, accuracy of detecting the stale state, duplicate action rate, average number of recovery attempts. Task success rate equals the number of successfully completed tasks divided by the total number of task executions, multiplied by 100. All executions have been recorded through structured logs that include task identification, type of injected failures, actions' times, inconsistent states detected, recovery actions performed and ultimate result of the process. Time required for recovery is considered as the period from failure detection till

Formula:

$$\text{Success Rate (\%)} = \frac{\text{Successful Tasks}}{\text{Total Tasks}} \times 100$$

successful task execution. The duplicate actions rate is estimated as the %age of executions where unintended repetitions occurred. Descriptive statistics, like %ages and averages, will be used for comparing results of the strategies under different failure conditions.

The experiment is based on simulated data and failure injections, not on a production environment. As such, results should be regarded as benchmark data that reflect the performance under particular conditions but not actual performance of the strategy. Replicability of the results will be ensured by keeping consistent task inputs, failure scenarios, evaluation criteria and recovery policies. This approach allows building a framework for detecting those recovery actions that increase the reliability of the tool.

Results

The results section will test the recoverability of the AI agents with partial failures and stale states through the simulation of 300 task executions through three recovery mechanisms: baseline execution, state-based recovery, and adaptive recovery. This is done through 100 executions for each of the three recovery mechanisms under four scenarios: normal, temporary failure of the tool, stale state, and combined failure. Calculation of task success ratio, recovery time, accuracy of detecting the stale state, duplicate actions, and number of recovery attempts will be calculated through explicit formulae (Singh, Saladi & Kumar, 2022). The numerical values shown below are purely simulated figures that represent the hypothetical results of an experiment not conducted.

Overall Task Completion Performance

The following sub-section will look into the capability of each of the recovery approaches in accomplishing the tasks that have been set out for them. This is due to the fact that task completion is the most important aspect since the agent should be able to recover from any form of interruption and accomplish its goal. These results are arrived at through the division of the total number of successful executions per total number of executions for each of the recovery approaches. There are 100 executions for each of the recovery approaches.

Table 1. Task Completion Outcomes Across Recovery Strategies

Recovery strategy	Successful tasks	Failed tasks	Total tasks	Success rate
Baseline execution	62	38	100	62%
State-aware recovery	79	21	100	79%
Adaptive recovery	90	10	100	90%

Calculation example — adaptive recovery:

$$\frac{90}{100} \times 100 = 90\%$$

Based on the example experiment results, adaptive recovery is 90 % successful in completing tasks, while state-aware recovery completes 79 % of tasks and baseline execution completes 62 % of tasks. In comparison to baseline execution, adaptive recovery increases task success by 28 % and by 11 % in comparison to state-aware recovery. It appears that using the combination of state checking, bounded re-attempts, checkpointing, and replanning might increase the ability of the agent to recover from failure as opposed to just trying the action again (Rengarajan *et al.* 2022). But these are not statistically proven findings; this can be done

through further experimentation with the collected data.

Performance Under Different Failure Conditions

The success rate in general may mask the variations in the types of failures. The following section discusses how the three recovery approaches behave under normal conditions, temporary tool failure, stale state, and the combination of all three failures. For each of these scenarios, there are 25 attempts for each approach, giving 100 attempts for each of the three approaches. The number of successes will then be divided by 25 to derive the success rates of each scenario.

Formula:

$$\text{Condition Success Rate (\%)} = \frac{\text{Successful Tasks in Condition}}{\text{Total Tasks in Condition}} \times 100$$

Table 2. Completion Rates Across Four Operational Scenarios

Failure condition	Baseline	State-aware	Adaptive
Normal operation	22/25 (88%)	23/25 (92%)	24/25 (96%)
Temporary tool failure	17/25 (68%)	21/25 (84%)	23/25 (92%)
Stale state	14/25 (56%)	19/25 (76%)	22/25 (88%)
Combined failure and stale state	9/25 (36%)	16/25 (64%)	21/25 (84%)
Total successful tasks	62/100	79/100	90/100

Formula:

$$\text{Condition Success Rate (\%)} = \frac{\text{Successful Tasks in Condition}}{\text{Total Tasks in Condition}} \times 100$$

Table 2. Completion Rates Across Four Operational Scenarios

Failure condition	Baseline	State-aware	Adaptive
Normal operation	22/25 (88%)	23/25 (92%)	24/25 (96%)
Temporary tool failure	17/25 (68%)	21/25 (84%)	23/25 (92%)
Stale state	14/25 (56%)	19/25 (76%)	22/25 (88%)
Combined failure and stale state	9/25 (36%)	16/25 (64%)	21/25 (84%)
Total successful tasks	62/100	79/100	90/100

calculation example — adaptive recovery under combined failure:

$$\frac{21}{25} \times 100 = 84\%$$

As we can see from the table above, the success rate of all strategies is at its lowest in the case of combined failures. Baseline execution has 36% success rate, while state-aware and adaptive recovery have 64% and 84%, correspondingly. The

Recovery Latency and Execution Time

Formula:

$$\text{Mean Recovery Time} = \frac{\sum_{i=1}^n t_i}{n}$$

Here, t_i is the recovery time for each successfully recovered execution, and (n) is the number of such executions with recorded timing data.

48-%age point gap between adaptive and baseline recovery in the presence of combined failures indicates the usefulness of such operations in the case of simultaneous occurrence of failures. However, these are simulated results, and further tests are required.

Table 3. Time Required to Restore Successful Execution

Recovery strategy	Successful recoveries with timing data	Total recovery time (seconds)	Mean recovery time
Baseline execution	62	1,140.8	18.4 seconds
State-aware recovery	79	1,121.8	14.2 seconds
Adaptive recovery	90	1,044.0	11.6 seconds

Calculation example — adaptive recovery:

$$\frac{1,044.0}{90} = 11.6 \text{ seconds}$$

The mean recovery time of adaptive recovery is 11.6 seconds, as opposed to 14.2 seconds for state-aware recovery and 18.4 seconds for the baseline. The %

time savings in relation to the baseline is about 37.0%.

$$\text{Time Reduction (\%)} = \frac{18.4 - 11.6}{18.4} \times 100 = 36.96$$

This would mean that any recovery method which includes validation and planning by the state could lower the delay. Nevertheless, latency alone is not enough to determine the reliability of the method; rather, the success rate of the tasks from Tables 1 and 2 should be taken into account along with latency.

Stale-State Detection Accuracy

Detection of stale state is an essential aspect of recovery of the AI agents since use of stale information may lead to erroneous decision making and failure to complete the assigned task. In this regard, this sub-section will make a comparison between the capability of the 3 recovery approaches in terms of detecting stale information from 50 affected executions per each approach.

Table 4. Detection Performance Across Recovery Strategies

Recovery strategy	Correct detections	Missed detections	Total cases	Detection accuracy
Baseline execution	22	28	50	44%
State-aware recovery	41	9	50	82%
Adaptive recovery	47	3	50	94%

Dis Results have shown that the accuracy of detection in case of adaptive recovery is the highest at 94%, then the state-aware recovery comes second at 82% and lastly the baseline execution at 44%. The adaptive recovery detects 47 of the 50 instances of the stale states correctly, while the baseline execution fails to detect 28 instances. This may show that the verification of the states may be useful for agents to detect the stale information before acting on it. The values above, however, only show simulation data and not any actual testing.cussion.

Frequency of Duplicate Actions

Duplication of actions is a process whereby an artificial intelligence performs a particular action twice without having to fail first or use an action before doing another one. This kind of behavior might be caused by a delay in tool responses, failure to log the performance of actions, or insufficient verification of states. The following section will discuss the comparison of duplicate action frequency for the 3 methods of recovery.

Table 5. Incidence of Unintended Repeated Operations

Recovery strategy	Executions with duplicate actions	Total executions	Duplicate-action rate
Baseline execution	18	100	18%
State-aware recovery	9	100	9%
Adaptive recovery	4	100	4%

Adaptive recovery has the minimum duplication rate of only 4%, followed by 9% for state-aware recovery and 18% for baseline execution. That implies that there has been a decrease in the duplication rate by 14% points relative to the baseline execution. However, the implication here is that verification of whether the action has been successful before proceeding will minimize the duplication rate. This is especially significant when dealing with agents interacting with external systems, because repetition in this case could lead to

inconsistency in records or other unforeseen consequences.

Recovery Attempts per Task

Efforts to recover demonstrate the frequency with which an agent needs to attempt other actions in order to complete the task after a failure. Such efforts can make the action take more time to execute, use computer resources, and make demands on other tools. This sub-section compares the average number of recovery efforts made by all three methods with one hundred executions of each.

Table 6. Additional Attempts Required During Recovery

Recovery strategy	Total recovery attempts	Task executions	Mean attempts per execution
Baseline execution	180	100	1.80
State-aware recovery	150	100	1.50
Adaptive recovery	130	100	1.30

Recovery via adaptive strategy achieves the minimum average attempt rate at 1.30 attempts per execution, in contrast to 1.50 and 1.80 attempts per execution in state-aware and baseline strategies respectively. The findings imply that verification and replanning of the state of the agents may contribute to the process of selecting more suitable strategies rather than repeating the same action. In addition, decrease in the number of attempts might lead to a decrease in tool usage and efficiency of the process. Nevertheless, low numbers of attempts cannot mean better performance when the tasks are not completed.

Comparative Recovery Efficiency

The combination of different performance indicators enables the assessment of the AI agent's recovery from failure in a more general way. The success rate indicator evaluates the extent to which the goals are accomplished, the recovery time assesses the delay that is induced by the failures, the stale-state detection evaluates the state awareness, and the duplicate action rate estimates the amount of unintended repetitions.

Table 7. Multi-Metric Comparison of Recovery Strategies

Evaluation metric	Baseline	State-aware	Adaptive
Task success rate	62%	79%	90%
Mean recovery time	18.4 seconds	14.2 seconds	11.6 seconds
Stale-state detection accuracy	44%	82%	94%
Duplicate-action rate	18%	9%	4%
Mean recovery attempts	1.80	1.50	1.30

From this analysis, we can see that the best illustrative performance is achieved by adaptive recovery in all five factors measured. This method has the highest success rate of task completion and accuracy of stale state detection along with the shortest recovery time and lowest duplicate action rates and average number of recovery attempts. The state-aware recovery method has better performance than baseline execution in these factors and could prove beneficial for adding state validation to the recovery process. From this result, one could hypothesize that the integration of state validation, bounded retrying, restoring checkpoint, and replanning may be beneficial. Nevertheless, this

analysis uses simulated numbers and does not have a statistical proof yet.

Improvement Relative to Baseline Execution

The current section highlights the differences between adaptive recovery and baseline execution to illustrate the possible advantages of the suggested recovery approach. The analysis considers aspects of task completion, detection of the stale state, recovery duration, duplication of actions, and recovery attempts. It is better to have higher levels of success and detection, while it is preferable to have shorter recovery time and less duplicate actions and attempts.

Table 8. Performance Improvements Achieved by Adaptive Recovery

Performance metric	Baseline execution	Adaptive recovery	Difference
Task success rate	62%	90%	+28 %age points
Stale-state detection accuracy	44%	94%	+50 %age points
Mean recovery time	18.4 seconds	11.6 seconds	6.8 seconds lower
Duplicate-action rate	18%	4%	14 %age points lower
Mean recovery attempts	1.80	1.30	0.50 fewer attempts

Adaptive recovery shows the greatest increase in stale-state detection accuracy, from 44% to 94%. Task success is also increased from 62% to 90%, while average recovery time is reduced by 6.8 seconds. Duplicate action ratio is reduced from 18% to 4%, indicating that execution state checking might help avoid duplicate actions. Thus, in total, the above examples can be used to show that recovery technique performance needs to be analyzed based on a number of criteria, not just task success (Zhang *et al.*, 2021). However, note that the figures mentioned are only example values obtained via simulation, not empirical results. Further experimentation needs to repeat trials and record agent activity logs.

Discussion

The results demonstrate the possible relevance of adaptive recovery systems to increasing the reliability of AI agents during partial failure situations and state staleness. The simulation of task performance resulted in 90% success rate for adaptive recovery, whereas state-aware recovery had a 79% success rate, and baseline execution, 62%. It could be hypothesized that the combination of state checking, checkpoint recovery, limited number of attempts, and replanning could increase the chances of successful task performance during disruptions. The results agree with other works on the topic of large language model agents, which emphasize planning, memory, and action capabilities (Alsalem *et al.*, 2018). Stale state detection was another important aspect. Adaptive recovery had a 94% success rate of detecting stale states, while baseline execution had 44%. Improved detection of stale states might avoid having agents perform operations on stale information resulting in errors. Moreover, the number of duplicate actions reduced from 18% with baseline execution to 4% under adaptive recovery. This shows how beneficial it is to check prior actions before performing any

operations. The shorter recovery period of 11.6 seconds is another benefit of adaptive recovery (Huang, Zhang & Zhang, 2020). Nevertheless, the results are simulated data, not experimental data. Therefore, the data does not prove that the adaptive recovery approach is better than any other methods in practice. In future research, these methods should be compared based on agent logs in practice, with multiple trials and different failures. Other factors, such as false positives, computation cost, and recovery cost, can also be included for a full evaluation of reliability.

Conclusion

This research study analyzed the recovery of an AI agent in case of partial failures and stale states via benchmarking the baseline execution, state-based recovery, and adaptive recovery. Based on the illustration of the results, it is evident that the adaptive recovery has resulted in the highest success rates of task execution (90%) and state detection (94%), lowering the average recovery time to 11.6 seconds and minimizing the duplicate action rate to 4%. From these results, it is clear that there is a potential for improvement in the reliability and execution efficiency of an AI agent by implementing state verification, bounded number of retries, checkpoint recovery, and replanning. Nevertheless, the results are hypothetical in nature and not based on the outcome of the conducted experiment. Future studies should validate the developed benchmarking framework using real AI agents and conducting experiments in various conditions and trials.

Reference List

- [1] Alsalem, M. A., Zaidan, A. A., Zaidan, B. B., Hashim, M., Albahri, O. S., Albahri, A. S., ... & Mohammed, K. I. (2018). Systematic review of an automated multiclass detection and classification system for acute Leukaemia in terms of evaluation and benchmarking, open

- challenges, issues and methodological aspects. *Journal of medical systems*, 42(11), 204. <https://www.academia.edu/download/93657491/s10916-018-1064-920221104-1-ruqu9f.pdf>
- [2] Bartolini, A., Beneventi, F., Borghesi, A., Cesarini, D., Libri, A., Benini, L., & Cavazzoni, C. (2019, August). Paving the way toward energy-aware and automated datacentre. In *Workshop Proceedings of the 48th International Conference on Parallel Processing* (pp. 1-8). <https://dl.acm.org/doi/pdf/10.1145/3339186.3339215>
- [3] Huang, L., Zhang, C., & Zhang, H. (2020). Self-adaptive training: beyond empirical risk minimization. *Advances in neural information processing systems*, 33, 19365-19376. https://proceedings.neurips.cc/paper_files/paper/2020/hash/e0ab531ec312161511493b002f9be2ee-Abstract.html
- [4] Hutchinson, B., Rostamzadeh, N., Greer, C., Heller, K., & Prabhakaran, V. (2022, June). Evaluation gaps in machine learning practice. In *Proceedings of the 2022 ACM conference on fairness, accountability, and transparency* (pp. 1859-1876). <https://dl.acm.org/doi/pdf/10.1145/3531146.3533233>
- [5] Rengarajan, A. (2022). Developing AI Powered Enterprise Integration Platforms Through Cloud Native Observability and Automated Governance. *International Journal of Science, Research and Technology*, 5(6), 8970-8977. <http://www.ijsrat.com/index.php/ijsrat/article/download/216/208>
- [6] Singh, N., Saladi, S. P., & Kumar, M. (2022). Automated Validation of Transaction Consistency and Recovery in Distributed Payment Systems: Duplicate Transactions, Retries, Timeouts, and Recovery. *INTERNATIONAL JOURNAL OF ADVANCES IN SIGNAL AND IMAGE SCIENCES*, 19-33. <http://www.xlescience.org/index.php/IJASIS/article/view/2187>
- [7] Yadav, D., Jain, R., Agrawal, H., Chattopadhyay, P., Singh, T., Jain, A., ... & Batra, D. (2019). Evalai: Towards better evaluation systems for ai agents. *arXiv preprint arXiv:1902.03570*. <https://arxiv.org/pdf/1902.03570>
- [8] Zhang, Q., Schulze, M., Gazdzicki, P., & Friedrich, K. A. (2021). Comparison of different performance recovery procedures for polymer electrolyte membrane fuel cells. *Applied Energy*, 302, 117490. <https://www.sciencedirect.com/science/article/pii/S0306261921008771>